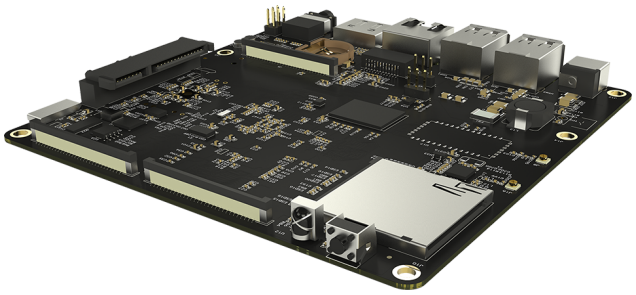
LOFT-Q 快速指南

组织: Mixtile Team
编写: Phil.Han <phil.han@focalcrest.com>
版本: 0.1
日期: 2015.01.23



LOFT-Q 是 Mixtile 项目的第二代原型板，基于全志 A31 芯片，主要面向嵌入式开发人员，工程师，创客，以及极客，可以作为家庭影音中心，个人云存储设备，NAS，等。该指南主要用于向开发人员介绍如何快速的根据自己的需要搭建合适的开发环境，并基于 LOFT-Q 定制自己的应用。

环境准备

在进行正式的 LOFT-Q 使用之前，如果在板子中已经烧录了最新稳定的系统镜像，则可以直接使用。如果需要根据自己的需要对硬件进行重新配置，或者添加一些硬件接口，需要重新构建系统镜像，需要执行如下的准备工作。

LOFT-Q 代码

LOFT-Q 项目目前的代码主要包含一下几个部分：

- build: https://github.com/mixtile/loftq-build
- uboot: https://github.com/mixtile/loftq-uboot
- linux: https://github.com/mixtile/loftq-linux
- buildroot： https://github.com/mixtile/buildroot
- android： 代码量比较大，后续打包提供。

后续，我们会提供更多的有关新系统移植的进展及下载地址。

代码下载

您可以通过下述命令来下载各个部分的代码，并且完成构建环境搭建。

- 创建构建目录

  通过下面的指令创建用于 LOFT-Q 项目的目录：

```
mkdir loftq
cd loftq
```

- 下载代码

  您可以执行下述命令来下载各个部分代码, 对于 linux 内核和 buildroot 部分, 代码量比较大, 可能需要比较长的时间才能完成下载。

```
git clone https://github.com/mixtile/loftq-build.git
git clone https://github.com/mixtile/loftq-uboot.git
git clone https://github.com/mixtile/loftq-linux.git
git clone https://github.com/mixtile/buildroot.git
```

关于 loftq-build

loftq-build 包含用于完成 uboot， linux , android, 以及后续添加代码的构建指令, 以及 sunxi 的一部分打包相关工具。

在执行具体的构建指令之前，我们需要执行如下指令，在当前的命令行环境下导入相关的构建指令：

```
source loftq-build/sunxi_env.sh
```

之后，我们可以执行后续的构建指令，下述是构建 linux 版 uboot 的指令：

```
linux_build_uboot
```

更多说明

sunxi_env.sh 类似于 android 的 lunch 配置文件。用于向当前终端中导入可用环境变量以及一些指令。

sunxi_env.sh 的开头部分定义了 uboot, linux, 以及 buildroot, android 等的路径，用户可以根据自己的需要进行修改，所有的可配置环境变量如下：

```
export BUILD_TRUNK=$(pwd)
export BUILD_TRUNK_OUT=$BUILD_TRUNK/out

# envs for sunxi tools
export SUNXI_TOOLS_PATH=$(pwd)/loftq-build
export SUNXI_LINUX_PATH=$(pwd)/loftq-linux
export SUNXI_UBOOT_PATH=$(pwd)/loftq-uboot
export SUNXI_TOOLCHAIN_PATH=${SUNXI_TOOLS_PATH}/toolschain/gcc-linaro/bin/

# envs for android
export ANDROID_TRUNK=$(pwd)/android
export ANDROID_DEVICE=loftq
export ANDROID_DEVICE_TRUNK=${ANDROID_TRUNK}/device/mixtile/${ANDROID_DEVICE}

# envs for ubuntu touch
# only used if we have android base sdk released by ubuntu touch team
# Note: now we can build this image but can't burn it to disk with PhoenixTool
export UBUNTU_OUTPUT=$BUILD_TRUNK_OUT/ubuntu
```

```
export UBUNTU_TARBALL=$UBUNTU_OUTPUT/vivid-preinstalled-touch-armhf.tar.gz

# commom env
export ANDROID_OUT=${ANDROID_TRUNK}/out
export ANDROID_DEVICE_OUT=${ANDROID_OUT}/target/product/${ANDROID_DEVICE}

export LINARO_GCC_PATH=$SUNXI_TOOLCHAIN_PATH
export PATH=$PATH:$LINARO_GCC_PATH
```

用户可以在其中添加自定义的指令，以 linux_build_uboot 为例：

```
function linux_build_uboot()
{
   CURDIR=$PWD

   cd $SUNXI_UBOOT_PATH
   make distclean
   make sun6i_config
   make -j4
   cd $CURDIR
}
```

可以参照上述格式进行添加。

UBoot 构建

基于 LOFT-Q 环境编译

目前可以使用 loftq-build 中的工具实现对 uboot 的快速编译。

对于构建适用于 Linux 和 Android 的构建指令分别如下：

- 构建适用于 Linux 的 Uboot.

  ```
  linux_build_uboot
  ```

- 构建适用于 Android 的 Uboot.

  ```
  android_build_uboot
  ```

手动定制编译

目前使用的 Uboot 除了可以使用构建工具进行编译之外, 还可以在 Uboot 目录下进行编译，相应的指令如下：

```
make distclean
make sun6i_config
make -j4
```

Linux 内核构建

对于全志的 linux 内核, 目前使用的是 linux 3.3 版本。目前这一版本的内核中统一了 linux 和 android 版本的内核。可以在同一份内核代码中完成对 Linux 和 android 内核的分开编译。

基于 LOFT-Q 环境的编译

目前可以使用 loftq-build 中的工具实现对 linux 内核的快速编译。下述指令分别实现对标准 linux 内核和 android 定制内核的编译。

- 标准 linux 内核编译

```
linux_build_kernel
```

- 定制 android 内核编译

```
android_build_kernel
```

手动定制编译

如果用户需要直接在 loftq-linux 中使用 linux 的 Makefile 进行内核的编译，需要完成如下工作。

- 导入 sunxi 的 linaro-gcc 编译工具链

```
export SUNXI_TOOLS_PATH=$(pwd)/loftq-build
export SUNXI_TOOLCHAIN_PATH=$SUNXI_TOOLS_PATH/toolschain/gcc-linaro/bin/
export PATH=$PATH:$SUNXI_TOOLCHAIN_PATH
```

对于上述的环境变量，需要根据自己的路径进行正确的配置。

- 标准 linux 内核编译

```
make distclean
./build.sh -p sun6i
```

- 定制 android 内核编译

```
make distclean
./build.sh -p sun6i_fiber
```

Buildroot 构建

Buildroot 是非常易于使用，定制，和构建的一套编译工具系统，便于开发人员快速的构建基于 Linux 内核的嵌入式操作系统。

获取源码

我们参照全志提供的较老的 buildroot 配置方法, 目前已经适配了最新的 buildroot 。目前开发人员可以从我们的 Github 项目代码库中下载最新的 buildroot 代码：

```
git clone https://github.com/mixtile/buildroot.git
```

我们会定期对 buildroot 进行更新以方便于使用最新的软件。

编译构建

在下载完 buildroot 源代码之后，可以使用下述命令完成对项目的编译和构建。

```
make mixtile_loftq_defconfig
make
```

> **Note**
>
> - 在执行 make 指令时, 将会从相关站点下载最新的代码, 同时在构建时将会编译生成很多临时文件, 因此应根据自己编译是选择的包的数量, 准备一定空余的磁盘空间, 否则可能会因为没有足够的空间而造成构建失败。
> - 在构建完成之后，将会在 output/images 目录中生成 rootfs.ext4 镜像，如果生成该文件，则表示构建成功。

镜像打包

在完成对 buildroot 的构建之后，将生成的 rootfs.ext4 文件拷贝到指定的目录，即 $BUILD_TRUNK/out/linux/。之后执行下述指令：

```
linux_pack
```

在执行完命令后将会打印出目标镜像的路径。则该文件就是 sunxi PhoenixTool 可以进行烧录的文件，可以烧录为卡启动，或者卡量产模式。

自我定制

如果您需要根据自己的需要，添加或者删除指定的软件库，可以参照如下说明进行定制。

```
make menuconfig
```

执行上述指令时，可能会提示缺少某些库，以至于无法完成操作，可以根据错误提示信息，安装制定的系统软件。

更多信息

- buildroot 官网: http://buildroot.uclibc.org
- buildroot 文档: http://buildroot.uclibc.org/docs.html

Ubuntu 构建

还有待添加

Android 构建使用

LOFT-Q 原型板的 Android 源码基于全志原厂提供的 Android 4.4.2 版本源码提供，增添了 LOFT-Q 相关的蓝牙, Wifi 部分驱动, 固件, 外接硬盘, USB, 红外遥控等的配置文件。由于 Android 源码过于庞大，我们提供了可供下载的压缩包，并未提供源码库，并且在源码中去除了提交日志等内容。

Android 的构建，同样可以使用类似于上述 Linux 和 buildroot 的基于 LOFT-Q 环境的编译和手动定制编译两种方式。

基于 LOFT-Q 环境构建 Android

基于 LOFT-Q 环境构建 Android 的指令如下：

1. 进入 android 源码目录

```
cd android
```

2. 构建 android 用 Uboot

```
android_build_uboot
```

3. 构建 android 用 Linux 内核

```
android_build_kernel
```

4. 构建 Android 项目

```
source build/envsetup.sh
lunch mars_loftq-eng
android_extract_bsp
make -j16
android_pack
```

OpenSUSE 使用

openSUSE 是来自德国的 Linux 发行版, 目前我们可以使用定制的 Uboot 和 Linux 内核来实现对 openSUSE ARM 版 JeOS 最简系统的引导和使用。

JeOS rootfs 下载

openSUSE 社区提供了多个版本的 JeOS rootfs 可供下载，如下：

- Factory 版本: http://download.opensuse.org/ports/armv7hl/factory/images/
- 13.1 版本: http://download.opensuse.org/ports/armv7hl/distribution/13.1/appliances/
- 12.3 版本: http://download.opensuse.org/ports/armv7hl/distribution/12.3/images/

我们可以从中选择一个版本的进行测试使用, 更多有关各个版本之间的异同信息, 可以参阅 https://en.opensuse.org/HCL:Chroot 。

以 13.1 版本的 openSUSE Jeos 根文件系统为例，我们需要下载名称如 openSUSE-*-ARM-JeOS.armv7-rootfs-*.tbz 的镜像文件。

生成 rootfs.ext4 文件系统

在下载完成 openSUSE 镜像文件之后，需要生成 rootfs.ext4 文件系统文件。具体步骤如下:

1. 参照 uboot 和 linux 内核构建说明，分别构建 Linux 系统用的 Uboot 和 内核。
2. 解压 openSUSE JeOS 镜像文件到目录 openSUSE-JeOS。
3. 执行下述指令生成 rootfs.ext4 文件系统文件。

```
./loftq-build/rootfs2ext4.sh -d ./openSUSE-JeOS -t ./rootfs.ext4
```

4. 将生成的 rootfs.ext4 文件拷贝到 out/linux/ 目录。

```
cp rootfs.ext4 ./out/linux/
```

5. 打包 Phoenix 工具用镜像文件。

```
linux_pack
```

> **Note**
>
> 需要根据提示, 输入相应选项, 打包完成后, 在 loftq-build/pack 目录下将会生成 sun6i_linux_loftq.img 文件。

6. 使用 Phoenix 工具烧录生成的 sun6i_linux_loftq.img 文件烧录到启动 TF 卡。

在完成上述步骤后, 将启动用的 TF 卡连接到 LOFT-Q, 即可使用 openSUSE JeOS 系统。